B: Amendments to The Claims:
Amend the claims to read as follows:

1   Claim 1. (Currently Amended) A multiprocessor computer
2   system, comprising:
3   a plurality of processing nodes and a plurality of
4   dynamic cache coherency regions using caches associated with
5   said processing nodes, and having
6   cache controller logic in said processing nodes
7   controlling ~~movement of~~ software_initiated movement of
8   software processes between said plurality of cache coherency
9   regions by changing coherency mode bits without requiring a
10   selective purging of cache contents in one or more of said
        processing nodes and when moving a software process between
1   two distinct sets of processing nodes a cache line cannot be
2   marked as shared in two separate coherency regions, and when
3   said supervisor program is moving a coherency region from
4   one distinct set of processing nodes to another distinct set
5   of processing nodes it is effectively leaving behind cache
6   entries for the coherency region on the old nodes and
7   ensures that these old cache entries will be seen by
8   incoming storage requests originating from the new
9   processing nodes and that cache entries for the same main
10   storage addresses will not be established in the new
11   processing nodes until the old entries are invalidated.

1   Claim 2. (Currently Amended) The multiprocessor computer
2   system according to claim 1, including supervisor software
3   which enables said cache controller logic in a processing
4   node to be sure upon an incoming storage request of a
        processor for a storage address that no copy of the
1   requested storage address exists outside that processor's

2   current coherency region, as specified by a ~~the~~ current
3   coherency region mode, whenever a cache entry for a
4   requested storage address is found to exist on any cache in
5   said processing nodes that contains ~~a~~ the processor that
6   initiated said incoming storage request.

1   Claim 3. (Original) The multiprocessor computer system
2   according to claim 2, wherein said supervisor software
3   creates a unique Coherency Region ID for each process that
4   has its own coherency region.

1   Claim 4. (Original) The multiprocessor computer system
2   according to claim 2, wherein supervisor software creates a
3   table for each processing node in the system which has an
4   entry for every Coherency Region ID that is currently
5   allowed to be dispatched on said processing node.

1   Claim 5. (Original) The multiprocessor computer system
2   according to claim 2, wherein said supervisor software
3   creates a unique Coherency Region ID for each process that
4   has its own coherency region and one or more coherency mode
5   bits for each processor in the multiprocessor computer
6   system, and said coherency mode bits and coherency region ID
7   associated with a processor are sent together with each
8   storage transaction that is initiated by that processor when
9   the transaction is transmitted for communication to another
10 processor of said multiprocessor computer system.

1   Claim 6. (Original) The multiprocessor computer system
2   according to claim 2, wherein said supervisor software
3   creates a unique Coherency Region ID for each process that
4   has its own coherency region and one or more coherency mode
5   bits for each processor in the multiprocessor system to

6   to a node controller for a processing node.

1   Claim 7. (Original) The multiprocessor computer system
2   according to claim 2, wherein said supervisor software
3   creates a unique Coherency Region ID for each process that
4   has its own coherency region and one or more coherency mode
5   bits for each processor in the multiprocessor computer
6   system, and wherein said mode bits associated with each
7   transaction are used to determine which caches must
8   participate in any storage transactions that they receive
9   from any of the processors of said multiprocessor computer
10   system.

1   Claim 8. (Original) The multiprocessor computer system
2   according to claim 2, wherein said supervisor software
3   creates a unique Coherency Region ID for each process that
4   has its own coherency region and one or more coherency mode
5   bits for each processor in the multiprocessor computer
6   system and enables multiple cache coherency regions to
7   operate without the use of cache purges during some
8   operations which move software processes between coherency
9   regions.

1   Claim 9. (Currently Amended) The multiprocessor computer
2   system according to claim 8, wherein said supervisor
3   software moves a software process out of one coherency
4   region that is no longer going to be used by said software
5   process and into another ~~software process~~ coherency region
6   that has been created to cover the same address space as the
7   first but which will include a new set of processing nodes.

1   Claim 10. (Original) The multiprocessor computer system
2   according to claim 2, wherein said supervisor software

3   creates a unique Coherency Region ID for each process that

4   has its own coherency region and moves a software process

5   from one coherency region encompassing one set of processing

6   nodes to another coherency region encompassing another set

7   of processing nodes without requiring cache purges of caches

8   in any of the processing nodes.

1   Claim 11. (Original) The multiprocessor computer system

2   according to claim 10, wherein if said another coherency

3   region contains fewer hardware processing nodes than the

4   original coherency region, the size of the coherency region

5   for said processing nodes has been effectively been reduced.

1   Claim 12. (Currently Amended) The multiprocessor computer

2   system according to claim 1, wherein said multiprocessor

3   computer system having a plurality of said ~~processsing~~

4   <u>processing</u> nodes uses a table of active coherency region

5   information associated with each processing node to

6   determine when to alter the processing nodes' cache state

     transitions.

1

2   Claim 13. (Original) The multiprocessor computer system

3   according to claim 12, wherein a supervisor program

4   initializes said tables associated with each processing node

5   and an entry in said table is made for each coherency region

6   that the supervisor program intends to use on that

     processing node.

1

2   Claim 14. (Original) The multiprocessor computer system

3   according to claim 1, wherein a supervisor program assigns a

4   unique coherency region ID for each coherency region which

5   the supervisor can associate with all software processes

6   that access storage addresses that are encompassed by the
7   coherency region.

1   Claim 15. (Original) The multiprocessor computer system
2   according to claim 1, wherein processing nodes are able to
3   identify incoming storage requests which target lines that
4   are no longer part of the address space of any software
5   process that is currently enabled by the supervisor software
6   to be dispatched on the node.

1   Claim 16. (Original) The multiprocessor computer system
2   according to claim 1, wherein processing nodes are able to
3   identify incoming storage requests which target lines that
3   are no longer part of the address space of any software
4   process that is currently enabled by the supervisor software
5   to be dispatched on the node to thereby identify cache lines
6   that are no longer actively used by any software processes
7   on that processing node and to change the cache entries for
8   that processing node to invalid in response to a storage
9   request from outside the coherency region.

1   Claim 17. (Original) The multiprocessor computer system
2   according to claim 1, wherein processing nodes are able to
3   identify incoming storage requests which target lines that
4   are no longer part of the address space of any software
5   process that is currently enabled by the supervisor software
6   to be dispatched on the node and allows all of the caches in
7   said multiprocessor computer system to continue processing
8   coherency transactions while the coherency boundaries for a
9   software process are effectively changed.

1   Claim 18. (Original) The multiprocessor computer system
2   according to claim 1, wherein processing nodes are able to

3   identify incoming storage requests which target lines that
4   are no longer part of the address space of any software
5   process that is currently enabled by the supervisor software
6   to be dispatched on the node such that cache lines belonging
7   to a software process that is no longer actively being
8   dispatched on a given processing node are identified and
9   invalidated thereby enabling their reuse.

1   Claim 19. (Original) The multiprocessor computer system
2   according to claim 1, wherein a supervisor software uses
3   processor state information to determine which caches in the
4   multiprocessor computer system are required to examine a
5   coherency transaction produced by a single originating
6   processor's incoming storage request.

1   Claim 20. (Currently Amended) The multiprocessor computer
2   system according to claim 19, wherein a processing node of
3   the multiprocessor computer system ~~has~~ has dynamic coherency
4   boundaries such that the multiprocessor computer system uses
5   only a subset of the total processors in a system for a
6   single workload at any specific point in time and can
7   optimize the cache coherency as the supervisor software
8   expands and contracts the number of processors which are
9   being used to run any single workload.

1   Claim 21. (Original) The multiprocessor computer system
2   according to claim 20, wherein multiple instances of
3   processing nodes can be connected with a second level
4   controller to create a large multiprocessor system.

1   Claim 22. (Original) The multiprocessor computer system
2   according to claim 21, wherein said supervisor software
3   creates a unique Coherency Region ID for each process that

4   has its own coherency region and one or more coherency mode

5   bits for each processor in the multiprocessor computer

6   system and enables multiple cache coherency regions to

7   operate without the use of cache purges during some

8   operations which move software processes between coherency

9   regions and a node controller uses said mode bits to

10   determine which processors must receive any given

11   transaction that is received by the node controller.

1   Claim 23. (Original) The multiprocessor computer system

2   according to claim 22, wherein a second level controller

3   uses the mode bits to determine which processing nodes must

4   receive any given transaction that is received by the second

5   level controller.

1   Claim 24. (Original) The multiprocessor computer system

2   according to claim 21, wherein said supervision software

3   uses logical partitions which are mapped to allowable

4   physical processors and a distinct cache coherency region

5   can be defined for each partition using a hypervisor.

1   Claim 25. (Original) The multiprocessor computer system

2   according to claim 2, wherein said coherency region ID is

3   used to perform the function of a cache coherency mode and a

4   node controller determines which physical processing nodes

5   are associated with specific coherency region Ids.

1   Claim 26. (Original) The multiprocessor computer system

2   according to claim 3, wherein any incoming storage request

3   which misses all of the caches in an originator's coherency

4   region is then sent on to all processing nodes in the entire

5   system, regardless of the setting of said mode bits.

1   Claim 27. (Currently Amended) The multiprocessor computer
2   system according to claim 3, wherein any incoming storage
3   request which requests ~~Requests~~ which hit in an originator's
4   coherency region but which do not have a correct cache state
5   do not need to be sent outside the coherency region.

1   Claim 28. (Currently Amended) A method for use in a
2   multiprocessor computer system, comprising the steps of:
3   moving software processes between a plurality of
4   cache coherency regions for caches associated with a
5   plurality of processing nodes of said multiprocessor
6   computer system without requiring a selective purging of
7   cache contents in one or more of said processing nodes,
8   after supervisor software creates a unique Coherency Region
9   ID for each process that has its own coherency region, and
10   said supervisor software creates a table for each
11   processing node in the ~~multiprocesor~~ multiprocessor computer
12   system which has an entry for every Coherency Region ID that
13   is currently allowed to be dispatched on said processing
     node.

1
2   Claim 29. (Original) The multiprocessor computer system
3   according to claim 28, wherein said coherency mode bits and
4   coherency region ID associated with a processor are sent
5   together with each storage transaction that is initiated by
6   that processor with a requested storage address when the
7   transaction is transmitted for communication to another
     processor of said multiprocessor computer system.

1
2   Claim 30. (Original) The multiprocessor computer system
3   according to claim 28, including a step of enabling with
4   supervisor software the multiprocessor computer system's
5   cache controller logic in a processing node to be sure that,

6   upon receipt at said processing node that an incoming

7   storage request for a storage address, no copy of the

8   requested storage address exists outside said processing

9   node's current coherency region, as specified by a current

10   coherency region mode, whenever a cache entry for a

11   requested storage address is found to exist on any cache in

12   any of said multiprocessor computer system's processing

13   nodes that contains a processor that initiated said incoming

     storage request.

1

2   Claim ~~27~~ 31. (RENUMBERED AND CANCELED) ~~The multiprocessor~~

3   ~~computer system according to claim 3, a cache line cannot be~~

4   ~~marked as shared in two separate coherency regions, and when~~

5   ~~said supervisor program is moving a coherency region from~~

6   ~~one distinct set of processing nodes to another distinct set~~

7   ~~of processing nodes it is effectively leaving behind cache~~

8   ~~entries for the coherency region on the old nodes and~~

9   ~~ensures that these old cache entries will be seen by~~

10   ~~incoming storage requests originating from the new~~

11   ~~processing nodes and that cache entries for the same main~~

12   ~~storage addresses will not be established in the new~~

     ~~processing nodes until the old entries are invalidated.~~